VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

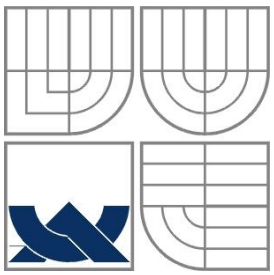
SYSTÉM PRO SPRÁVU VĚDECKÝCH PUBLIKACÍ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

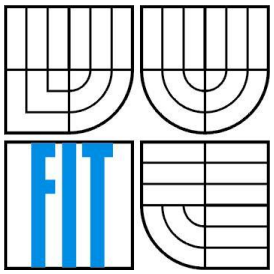
AUTOR PRÁCE
AUTHOR

JURAJ PAŠKA

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

SYSTÉM PRO SPRÁVU VĚDECKÝCH PUBLIKACÍ

SYSTEM FOR SCIENTIFIC PAPER MANAGEMENT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JURAJ PAŠKA

VEDOUCÍ PRÁCE
SUPERVISOR

ING. JAN KNĚŽÍK

BRNO 2012

Abstrakt

Cílem bakalářské práce je implementace víceuživatelského systému pro správu vědeckých publikací, který je schopný pracovat tak jak s místními úložišti dokumentů, tak se vzdáleným prostřednictvím protokolů HTTP a SFTP. Tyto dokumenty může systém vyhledávat a třídit podle zadaných klíčových slov, tak aj organizovat do skupin podle požadavek uživatelů. Výsledkem je multiplatformní aplikace implementovaná v jazyce Java.

Abstract

The main goal of the bachelor thesis is an implementation of multi-user system for administration of scientific publications which is able to handle local document storages as well as remote ones via HTTP and SFTP protocols. System can search and separate these documents based on the keyword input as well as organise them into groups per user specifications. Outcome of the thesis is a multiplatform application implemented in Java language.

Klíčová slova

správa vědeckých publikací, repositář, dokumentově orientovaná databáze, nerelační databáze, MongoDB, Java, Jython

Keywords

system for scientific paper management, repository, document-oriented database, non-relational database, MongoDB, Java, Jython

Citace

Paška Juraj: Systém pre správu vedeckých publikácií, bakalárska práca, Brno, FIT VUT v Brne, 2012

Systém pre správu vedeckých publikácií

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Jana Kněžíka
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Juraj Paška
16. mája 2012

Poděkování

Týmto by som chcel poďakovať predovšetkým svojmu vedúcemu práce Ing. Janovi Kněžíkovi za odbornú pomoc pri práci na tomto projekte a za cenné rady, ktoré mi poskytoval. Srdečná vďaka patrí taktiež mojej rodine, ktorá ma podporovala.

Obsah

1 Úvod

Hovorí sa, že žijeme v dobe informačného veku, informačnej spoločnosti a informačných technológií. V dobe, keď takmer každý má prístup k internetu, neobmedzenej studnici znalostí a informácií.

Mnohé z týchto znalostí sú s nami už od nepamäti a nahromadenie týchto informácií, vedomostí, múdrostí vyústilo k nutnosti ukladať ich napríklad v podobe zvitkov alebo kníh na jedno miesto, do knižníc. Tu sa o ne ľudia mohli starať, robiť medzi nimi poriadok, organizovať ich, ochraňovať ich, zdieľať ich medzi sebou. A takto to bolo po dlhé stáročia až do súčasnosti, ale doba pokročila a s rozvojom výpočtovej techniky a informačných technológií vznikli aj nové možnosti ukladania informácií v elektronickej podobe, pričom mnohé z nich sú roztrúsené po internete, no mnohé sú zhromaždené v nespočetných on-line databázach.

Všetci vieme, že dopracovať sa k požadovaným informáciám nie je vždy jednoduché a o nič jednoduchšie nie je v nich sa zorientovať. Keďže ich väčšinou nenájdeme na jednom mieste, na jednej stránke, v jednej databáze, zvykneme si nájdené dáta ukladať k sebe do osobného počítača, kde sa v nich časom, ako sa postupne dáta hromadia, začneme strácať. Riešením tohto problému sú aplikácie, ktoré nám uľahčujú správu a organizáciu dát. Jedným z týchto typov aplikácií sú programy pre správu vedeckých publikácií. A práve s týmito aplikáciami a problémami s nimi spojenými sa budem zaoberať v tejto práci.

1.1 Štruktúra bakalárskej práce

Cieľom projektu je vypracovať analýzu existujúcich aplikácií pre správu vedeckých publikácií a najmä navrhnúť viac užívateľský systém pre správu vedeckých publikácií, ktorý bude schopný pracovať ako s miestnym úložiskom dokumentov, tak aj so vzdialeným pomocou protokolov HTTP a SFTP. Následne je nutné tento návrh implementovať a vytvoriť klienta spolu so vzorom lokálneho i vzdialeného úložiska dokumentov.

Teoretická časť jednoducho a stručne pojednáva o technológiách použitých pri implementácii aplikácie. Oboznamuje čitateľa o databáze MongoDB, jej možnostiach a výhodách. Popisuje formát súboru JSON a taktiež sa venuje programovaciemu jazyku Java, programovaciemu jazyku Jython a ich vzájomnej spolupráci.

Ďalšia kapitola vysvetľuje pojem repozitár, rozoberá možné funkcie systémov pre správu vedeckých publikácií a porovnáva tri takéto aplikácie: Mendeley, Papers a I,Librarian.

V nasledujúcej kapitole sa budem zaoberať návrhom jednotlivých častí systému. Predstavím formu ukladania dát na lokálne a vzdialené úložisko, štruktúru dát v databáze, spôsob prenosu dát medzi klientskou aplikáciou a serverom. Taktiež predstavím počiatočný návrh užívateľského rozhrania.

Piata kapitola sa venuje už samotnej implementácii programu. Popisuje užívateľské rozhranie, adresárovú štruktúru servera a obsah súborov, ktoré sú na ňom uložené. Taktiež priblíži princíp fungovania jednotlivých funkcií aplikácie.

Predposledná kapitola sa zoberá testovaním aplikácie po funkčnej a grafickej stránke. V závere je zhrnutie a zhodnotenie celej práce.

2 Teoretický rozbor

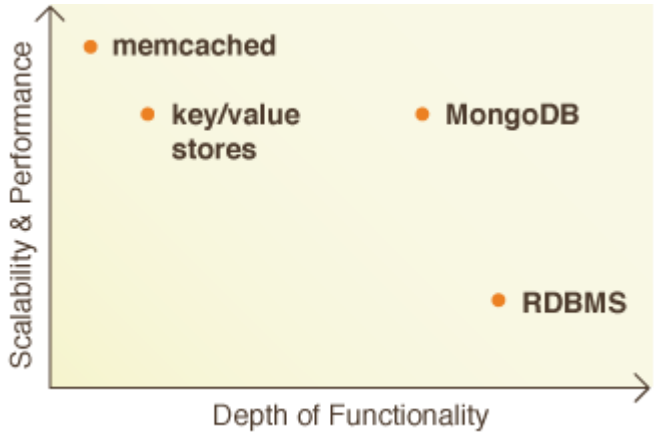
Cieľom tejto kapitoly je predstavenie jednotlivých technológií použitých pri implementácii aplikácie. Pojednáva o databáze MongoDB, o formáte súborov JSON, o programovacích jazykoch Java a Jython. Tiež nám stručne predstaví protokol SFTP.

2.1 MongoDB

MongoDB je dokumentovo orientovaná nerelačná databáza so zameraním na flexibilné ukladanie dát a jednoduchosť použitia. Patrí do skupiny takzvaných NoSQL databáz. Pomenovanie tejto skupiny vychádza z toho, že má vlastný dotazovací jazyk a nie SQL. Je napísaná v C++ a od roku 2009 vyvíjaná ako open source projekt pod licenciou AGPL [1].

2.1.1 Vlastnosti MongoDB

MongoDB je dobre škálovateľná, s vysokým výkonom a s funkcionalitou na úrovni relačných databáz, ako sú napríklad sekundárne indexy, dotazy s obmedzením rozsahu (range query) a triedenie [2]. Porovnanie je znázornené na obrázku 2.1, z ktorého môžeme vidieť, že pri výkone takmer na úrovni systémov memcached[1] si MongoDB zachováva funkcionalitu podobnú databázam typu



Obrázok 2.1: Porovnanie výkonu a funkcionality niekoľkých typov databázových systémov[4].

RDBMS[2]. Samozrejme žiadna databáza nie je ideálna a nie je riešením všetkých problémov. MongoDB sa však pokúša vytvoriť čo najoptimálnejšie riešenie, i keď preto občas musí obetovať určité veci ako detailná kontrola a ladenie, príliš náročnú funkcionalitu, ktorá vyžaduje množstvo

[1] Memcached je univerzálny systém pre cachovanie dát do pamäti [3]
[2] RDBMS (relational database management system) - relačné databázy

komplikovaného kódu a logiky na aplikačnej vrstve a niektoré ACID[3] vlastnosti ako multi-dokumentové transakcie. Medzi jej výhody patrí [2][5]:

2.1.1.1 Flexibilný a bohatý dátový model

Oproti relačným databázam, kde je každý záznam chápaný ako „riadok“, u MongoDB je každý záznam chápaný ako „dokument“. Možnosť zanorovania sa dokumentov a polí umožňuje dokumentovo orientovanému prístupu databázy reprezentovať zložité hierarchické vzťahy prostredníctvom jediného záznamu. Táto vlastnosť je preto výhodou pre vývojárov v moderných objektovo orientovaných jazykoch, keďže môžu uložiť ich objekt, hoci i zložitejší do jedného záznamu. MongoDB je bezschémovou databázou. To znamená, že kľúče dokumentu nie sú pevne preddefinované a preto každý záznam môže obsahovať rozdielne dáta a nemusí obsahovať pevne stanovenú formu.

2.1.1.2 Jednoduché škálovanie

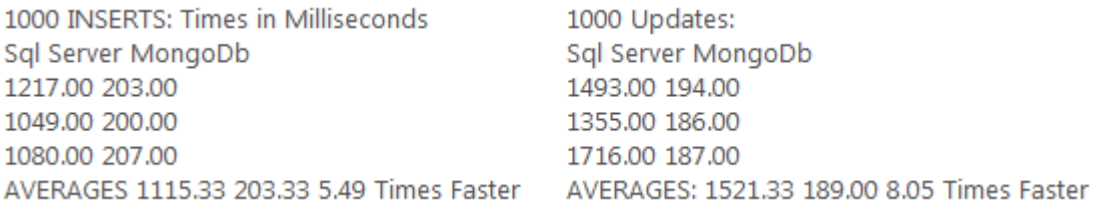
V súčasnej dobe vďaka rozširovaniu informačných technológií výrazne rastie množstvo dát ukladaných na serveroch do databáz. Tento problém má v podstate dve riešenia. Prvou možnosťou zvýšenia kapacity alebo výkonu je vylepšenie serveru kúpou výkonnejšieho hardware, čo nie je síce zložité, ale môže to byť dosť drahé a aj hardware ma svoje fyzické limity, ktoré už nemôže prekročiť. Druhou možnosťou zvýšenia kapacity alebo výkonu je kúpenie ďalšieho serveru a pridanie tohto serveru do clusteru[4]. Práve pre túto možnosť je MongoDB optimalizovaná. Jej objektovo orientovaný dátový model jej umožňuje jednoducho prerozdeliť dáta medzi viacero serverov, dokonca bez akéhokoľvek reštartovania systému.

2.1.1.3 Rýchlosť

Je všeobecne známe, že dokumentovo orientované databázy sú rýchlejšie ako relačné databázy. Vďaka držaniu dát pokope v dokumentoch môžu byť dotazy na databázu oveľa rýchlejšie ako pri relačných databázach, kde sa dáta musia vyhľadávať vo viacerých tabuľkách a nakoniec spojiť. Rýchlosť ovplyvňuje takisto to, že MongoDB používa na ukladanie a prenášanie dát po sieti binárny protokol vo formáte BSON[5]. Tiež využíva mapovanie súborov do pamäti, čím prenecháva zodpovednosť za správu pamäte operačnému systému.

Ukážku amatérskych testov je možné vidieť na obrázku 2.2

[3]ACID je súbor vlastností, ktoré zaručujú, že databázové transakcie prebehnú spoľahlivo
[4] Cluster je skupina spolupracujúcich počítačov, pričom navonok vystupujú ako jeden počítač.
[5] BSON (Binary JSON (viac v kapitole 2.2)) je binárny formát, v ktorom je žiadny alebo viac párov kľúč/hodnota uložených v jedinej entite nazývanej dokument [3].



Obrázok 2.2: Porovnanie rýchlosti SQL a MongoDB (amatérsky test) [6][7]

2.1.1.4 Množstvo funkcií

MongoDB podporuje množstvo funkcií oboch typov databáz, tak ako tradičných relačných, tak aj dokumentovo orientovaných. Sú to napríklad:

- sekundárne indexy
- vykonávanie JavaScriptu na strane servera
- pevná veľkosť kolekcií[6], čo je užitočné napríklad pre logovanie dát
- dynamická replikácia dát medzi servermi

Medzi výhody MongoDB určite patrí množstvo podporovaných programovacích jazykov a množstvo administračných aplikácií.

2.2 JSON

JSON (JavaScript Object Notation) je textový dátový formát určený pre výmenu dát [8]. Vychádza z programovacieho jazyka JavaScript, no na programovacom jazyku je plne nezávislý. V základe používa kódovanie UTF-8. Základné dátové typy JSON sú:

- číslo (má podobný formát ako v Jave, ale podporuje len dekadickú sústavu)
- reťazec (postupnosť znakov v úvodzovkách `""`)
- boolean (`true` alebo `false`)
- pole (zoznam hodnôt oddelených čiarkou uzavretý v hranatých zátvorkách `[]`)
- objekt (množina párov `názov:hodnota` oddeľovaná čiarkami a uzavretá v zložených zátvorkách `{}`)
- `null`

Je považovaný za odľahčenú alternatívu k formátu XML. Príklad JSON je znázornený na obrázku 2.3 a Jeho ekvivalentný zápis v XML na obrázku 2.4.

[6] Kolekcia v MongoDB je pomenovaná skupina dokumentov. Zhruba tabuľka v relačných databázach.

```json
{
        "number": 10,
        "string": "Foo",
        "boolean": true,
        "array": ["Bar","Eek"],
        "object": { "number":25.5, "other":null }
}
```

Obrázok 2.3 Ukážka JSON

```xml
<example>
    <number>10</number>
    <string>Foo</string>
    <boolean>true</boolean>
    <array>
        <item value="Bar" />
        <item value="Eek" />
    </array>
    <object>
        <number value="25.5" />
        <other value="null" />
    </object>
</example>
```

Obrázok 2.4 Ukážka XML

2.3 Java

Java je objektovo orientovaný programovací jazyk predstavený v roku 1995 firmou Sun Microsystems, no jej vývoj začal už v roku 1991. Je založená na princípoch jazykov C a C++, ale je postavená takmer úplne na objektoch. V súčasnosti je jedným z najpoužívanejších jazykov na svete.

Medzi jej základné charakteristiky patrí prenositeľnosť, čo znamená, že program napísaný v Jave funguje na hocijakom hardware a operačnom systéme, ktorý má k dispozícii interpret Javy – Java Virtual Machine (JVM) . Táto vlastnosť je dosiahnutá tým, že preklad neprebieha do strojového jazyka počítača, ale do pseudojazyka nazývaného Java bytecode. Tento jazyk je nezávislý od cieľového počítača. Preložený program – bytecode – je uložený v súboroch s príponou `.class`, ktoré sú po spustení programu zavádzané do pamäti, pričom prebieha overovanie bytecodu. Po overení je program spustený pomocou interpreta [9].

Medzi ďalšie vlastnosti Javy patria jednoduchosť (vychádza z jazyka C++, ale odpadla väčšina konštrukcií, ktoré robili programátorom problémy), robustnosť (používa silnú typovú kontrolu),

a správa pamäte. Správa pamäte je realizovaná pomocou automatického Garbage collectoru, ktorý automaticky uvoľňuje pamäť po objektoch, na ktoré už neexistuje odkaz. To neznamená, že v Jave neexistuje niečo ako únik pamäti[7]. Niečo podobné môže nastať, ak sa v programe nachádza odkaz na objekt, ktorý sa viac nebude používať.

Medzi hlavné nevýhody Javy sa považuje rýchlosť štartu aplikácií. Keďže Java je jazyk interpretovaný (prekladá sa najprv do bytecode), nemôže dosahovať rýchlosti jazykov prekladaných priamo do strojového kódu. Ďalšou negatívnou vlastnosťou, najmä pri menších programoch, je pamäťová náročnosť spôsobená nutnosťou mať v pamäti celé prostredie aplikácie [10].

2.4 Jython

Jython je kompletná implementácia jazyka Python[8] pre platformu Java. Vďaka tomu môže Python bežať na akomkoľvek zariadení, na ktorom beží Java Virtual Machine. Je to kompletný programovací jazyk, ktorý umožňuje písanie aplikácií čisto v Jython [11]. Obsahuje takmer všetky moduly štandardnej distribúcie jazyka Python (chýbajú iba moduly pôvodne implementované v C).

2.4.1 Spolupráca s Javou

Jython taktiež umožňuje importovanie knižníc a balíčkov jazyka Java, čo výrazne rozširuje jeho možnosti. Vďaka tomu je možné v Jython skriptoch vytvárať objekty z importovaných knižníc rovnakým spôsobom, ako sú vytvárané v Python. Treba však poznamenať, že importovaná knižnica musí byť zahrnutá v Java CLASSPATH[9]. Následne je možné volať metódy a funkcie importovaných tried rovnakým spôsobom ako v Python. Mapovanie dátových typov jazyka Java a jazyka Python je možné vidieť na obrázku 2.6.

Java Type	Python Type
char	String(length of 1)
boolean	Integer(true = not zero)
byte, short, int, long	Integer
java.lang.String, byte[], char[]	String
java.lang.Class	JavaClass
Foo[]	Array(containing objects of class or subclass of Foo)
java.lang.Object	String
orb.python.core.PyObject	Unchanged
Foo	JavaInstance representing Java class Foo

Obrázok 2.5 Dátové typy v Jave a Python [12]

[7] Únik pamäti je situácia, keď program alokuje pamäť a nie je schopný ju uvoľniť potom, čo ju už nepotrebuje.
[8] Python, občas označovaný aj ako CPython, je implementovaný v jazyku C
[9] Classpath je parameter Java programov, ktorý hovorí JVM, kde hľadať knižnice tretích strán alebo užívateľom definované triedy.

Jython je možné využívať v rámci Java aplikácií [13]. Jedným zo spôsobov je používanie objektu PythonInterpreter pre vykonávanie jednoduchých Python programov[13]. Druhým spôsobom, ktorý je dostupný od verzii Javy 6, je použitie ScriptEngineManageru [14]. Ten podporuje rôzne skriptovacie jazyky, podmienkou je, že musia mat skriptovací engine. Ako napríklad Jython, JavaScript, Ruby.

2.5 SFTP

SFTP (SSH File Transfer Protocol) je protokol zabezpečujúci bezpečný prenos súborov prostredníctvom počítačovej siete [15]. Pracuje na porte 22 na princípe žiadosť - odpoveď. Sám o sebe nezaisťuje autentizáciu, ani zabezpečenie prenášaných dát, ale používa k tomu iný protokol, zvyčajne SSH2[10]. Ponúka široké možnosti pre operácie s adresármi a súbormi po sieti, z nich tie najzákladnejšie sú napríklad:

- vytvorenie súboru
- odstránenie súboru
- vytvorenie adresára
- zmazanie adresára
- čítanie zo súboru
- výpis adresára
- zápis do súboru
- premenovanie súboru
- získanie vlastností súboru
- nastavenie vlastností súboru

[10] SSH2 je protokol pre bezpečné vzdialené prihlásenie a zabezpečenie ďalších služieb po sieti [16].

3 Analýza existujúcich aplikácií

V tejto kapitole by som chcel vysvetliť, čo je to repozitár, rozobrať možnosti aplikácií pracujúcich s vedeckými publikáciami a pozrieť sa podrobnejšie na už existujúce aplikácie pre správu vedeckých publikácií.

3.1 Čo je to repozitár

Vo všeobecnosti sa chápe ako miesto, kde sú dáta ukladané a organizovane spravované. Repozitár môže byť miesto, kde je uložených viacero databáz alebo súborov pre distribúciu po sieti alebo repozitár môže byť takisto miesto, ktoré je prístupné priamo pre užívateľa bez toho, aby k nim musel pristupovať cez sieť.

V akademickom prostredí sa repozitár chápe ako skutočné alebo virtuálne zariadenie pre uskladnenie vedeckých publikácií, ako napríklad vedeckých, novinových, časopisových článkov, elektronických kníh, diplomových prác a podobne.

K týmto repozitárom sa pripájajú aplikácie pre správu vedeckých publikácií. Štruktúra repozitára nie je určená nejakým pravidlom, ale závisí od implementácie aplikácie, ktorá k ním má primárny prístup. Vo všeobecnosti by mali repozitáre obsahovať aspoň samotné vedecké publikácie a databázu, ktorá si uchováva informácie o týchto dokumentoch.

3.2 Špecifikácia aplikácii

Úlohou aplikácie pre správu vedeckých publikácií je vo všeobecnosti uľahčiť užívateľovi spravovanie množstva dokumentov a dát, uľahčiť mu prístup k nim, zjednodušiť ich vyhľadávanie a organizáciu.

K základným vlastnostiam aplikácie by mala patriť možnosť ukladať dáta lokálne, tak aj na vzdialené úložisko pre umožnenie prístupu k svojím dokumentom z rôznych zariadení. Mala by podporovať základné operácie pre prácu s dokumentmi, ako vyhľadávanie, možnosť vytvoriť, upraviť, zmazať dokument. Taktiež by mala podporovať možnosť vyhľadávania v čo najväčšom množstve existujúcich internetových databáz, ktoré sú najväčším zdrojom vedeckých publikácií a následne nájdené dáta ukladať do vlastného repozitára. Formát ukladaných dát by nemal byť nijak obmedzený. Aplikácia by mala poskytnúť užívateľovi nejaký systém organizácie vlastných dát do skupín podľa jeho vlastného výberu.

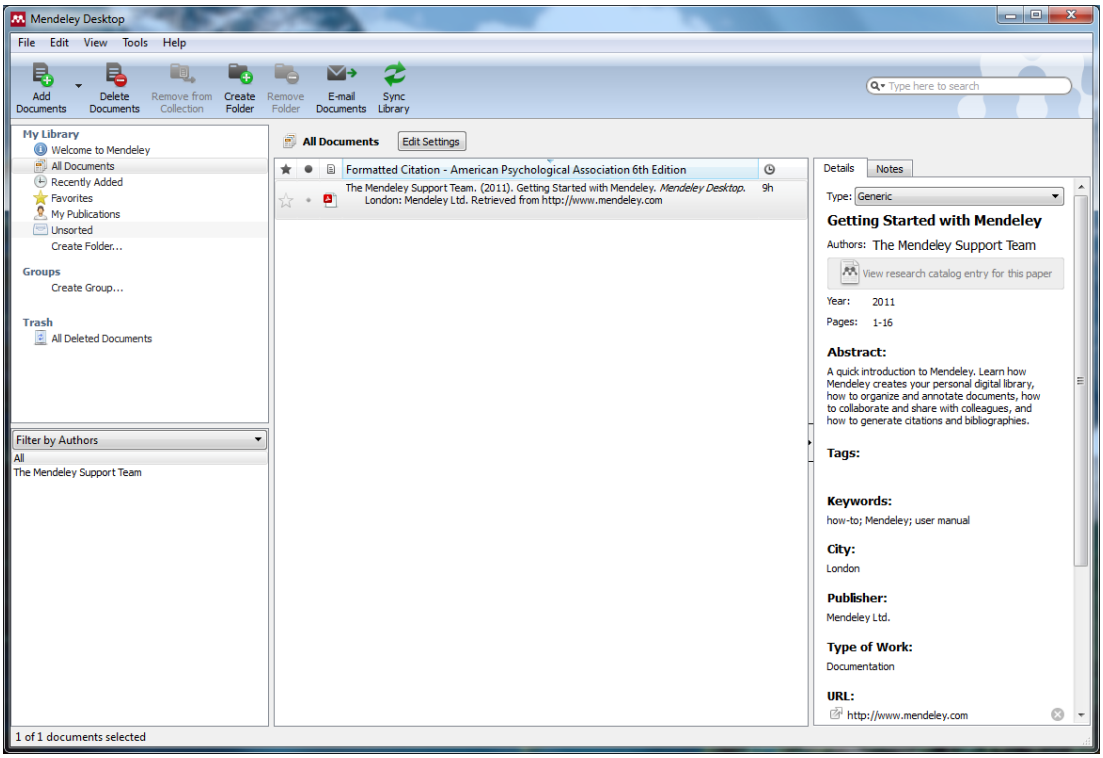
3.3 Rozbor aplikácií

3.3.1 Mendeley

Mendeley je multiplatformná aplikácia, bežiaca na operačných systémoch MS Windows, Linux a OS X. Taktiež umožňuje prístup aj prostredníctvom webového rozhrania.

Pre prácu s ňou je nutné si najprv vytvoriť účet. Tým získame ukladací priestor 1GB (v prípade príplatku aj viac) pre náš vlastný internetový repozitár. Zjednotenie lokálnej databázy s online repozitárom sa vykonáva manuálne. Mendeley poskytuje okrem základných operácií, možnosť odoslať e-mail s odkazom na dokument s vašej online databázy alebo generovať citácie (o dokumentoch) rôznych typov. Pri pridávaní dokumentov je možná extrakcia metadát. Je tu aj možnosť organizácie dokumentov do priečinkov a export záznamov.

Užívateľské rozhranie je jednoduché a prehľadné. V hornej časti okna sa nachádza lišta s rôznym menu, pod ňou sa nachádza panel s tlačidlami pre rýchly prístup k najpoužívanejším operáciám s vyhľadávacím oknom na pravej strane. Zvyšok okna je rozdelený na tri časti. Na ľavej strane je navigačný panel, centrálnu časť obrazovky tvorí panel, v ktorom sa zobrazujú výsledky vyhľadávania a v pravej časti je panel zobrazujúci podrobné informácie označených dokumentov.



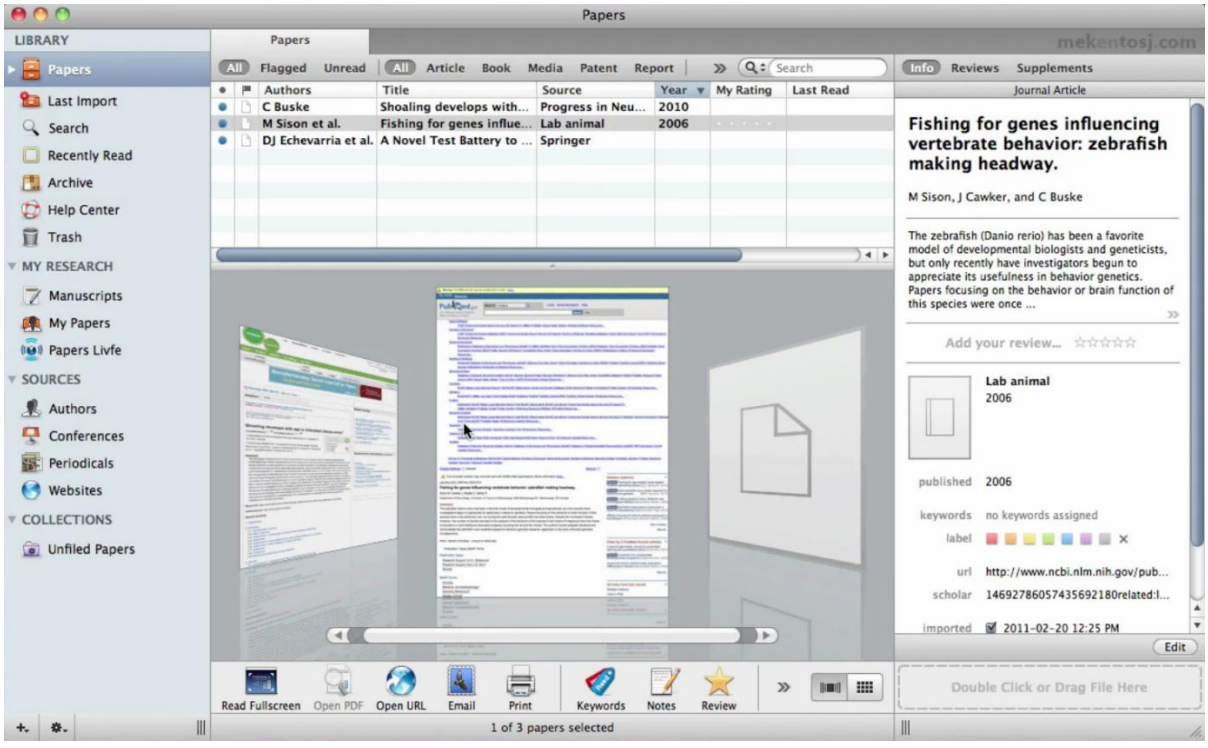
Obrázok 3.1 Ukážka aplikácie Mendeley Desktop

3.3.2 Papers

Papers je platená aplikácia, veľmi podobná vyššie analyzovanej aplikácií Mendeley, bežiaca na operačnom systéme OS X.

Pri prvom spustení aplikácie sa spustí sprievodca pre vytvorenie profilu a tým aj vlastnej databázy. Papers zvláda samozrejme všetky základné operácie. Dokáže odosielať vybraný dokument mailom, generovať citácie z dokumentov, extrahovať metadáta z dokumentov a organizovať dokumenty do zložiek pre lepšiu orientáciu. Existuje tiež možnosť pripájania doplnkov (rozumej súborov s doplňujúcimi informáciami) k existujúcim dokumentom. Taktiež je tu možnosť vyhľadávania v online databázach.

Užívateľské rozhranie je prehľadné. Je rozdelené do troch častí. V ľavej časti sa nachádza navigačný panel. Stredný panel môžeme rozdeliť do viac častí. V hornej časti sa nachádza lišta s rôznymi filtrami pre zobrazovanie dokumentov a vyhľadávacie okno. Pod ňou sa nachádza panel pre zobrazenie výsledkov vyhľadávania. Pod ním je panel s náhľadom na označený dokument a v dolnej časti sa nachádza lišta s tlačidlami pre rýchly prístup k vybraným operáciám. Pravú časť tvorí panel pre zobrazovanie podrobných informácií o vyznačenom dokumente.



Obrázok 3.2 Ukážka aplikácie Papers

3.3.3 I, Librarian

I, Librarian je aplikácia využívajúca webové rozhranie. Pre jej fungovanie je však potrebné mať nainštalované HTTP server Apache a PHP. Vďaka tomu môže bežať ako na MS Windows, tak aj na Linuxe a OS X.

Okrem základných operácií zvláda extrahovanie metadát z dokumentov, organizáciu dokumentov do kategórií, pripájanie doplnkov k existujúcim dokumentom a exportovanie záznamov v rôznych formátoch. Ďalej umožňuje priame vyhľadávanie v online databázach ako PubMed, PubMed Central, NASA ADS, arXiv, JSTOR® a HighWire Press® a import záznamov z ďalších online databáz vo formáte RIS.

Užívateľské rozhranie je rozdelené na dve časti. V ľavej hornej časti sa nachádza vyhľadávacie okno s možnosťou rozšíreného vyhľadávania. Pod ním sa nachádza zoznam položiek pre možnosť rýchleho filtrovania dokumentov databázy. Pravá časť slúži pre zobrazovanie výsledkov vyhľadávania, kde sa po kliknutí na nadpis vyhľadaného dokumentu presunieme na stránku s podrobným výpisom informácií o zvolenom dokumente.

Obrázok 3.3 Ukážka aplikácie I, Librarian

3.3.4 Súhrnné porovnanie

V nasledujúcej tabuľke je znázornené súhrnné porovnanie všetkých troch aplikácií. Ako je vidieť, rozdiely vo funkcionalite sú minimálne. Najväčšie rozdiely sú v používateľskom rozhraní a v jeho ovládaní.

	Mendeley	Papers	I, Librarian
Cena aplikácie	Zdarma	59 €	Zdarma
Lokálny úložný priestor	Yes	Yes	Yes
Online úložný priestor	Yes	No	No
Podpora Windows	Yes	Yes	Yes
Podpora OS X	Yes	Yes	Yes
Podpora Linux	Yes	No	Yes
Podpora mobilov / I-Pad	Yes	Yes	No
Typ rozhrania	Desktop, Web	Desktop	Web
Kompatibilita s najpoužívanejšími prehliadačmi	Yes	Yes	Yes
Základné operácie	Yes	Yes	Yes
Organizácia dokumentov	Yes	Yes	Yes
Generovanie citácií	Yes	Yes	No
Pokročilé vyhľadávanie	Yes	Yes	Yes
Vyhľadávanie v internetových databázach	No	Yes	Yes
Extrakcia metadát z PDF	Yes	Yes	Yes
Export informácií o dokumente	Yes	Yes	Yes

Obrázok 3.4 Porovnanie aplikácií pre správu vedeckých publikácií

4 Návrh riešenia

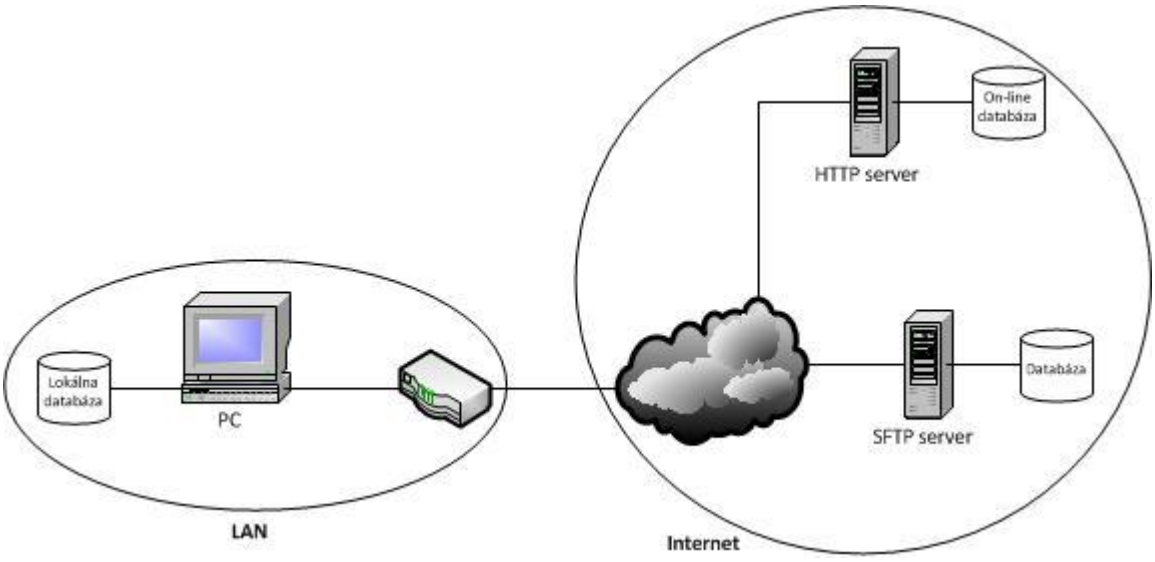
Obsahom tejto kapitoly je návrh systému ako celku a návrh štruktúry databázy. Popíšem, ako by mala vyzerať práca aplikácie so vzdialenými servermi, tak aj so servermi umiestnenými na počítači s klientskou aplikáciou.

4.1 Celkový návrh systému

Systém (znázornený na obrázku 4.1) bude pozostávať z dvoch základných častí, a to klientskej aplikácie s grafickým rozhraním a zo serverovej časti, ktorú predstavujú repozitáre. Program bude na komunikáciu zo vzdialenými úložiskami (repozitármi) využívať už existujúce protokoly, a to SFTP (2.5) alebo HTTP. Podľa typu pripojenia môžeme servery rozdeliť na tri základné typy:

- SFTP server
- HTTP server
- lokálny server

HTTP protokol je určený na komunikáciu s internetovými databázami. O správu dát na lokálnom úložisku a na SFTP repozitári sa bude starať priamo klientská aplikácia, preto nie je potrebné vytvárať serverovú časť systému a je možné využiť už existujúce serverové aplikácie pracujúce s týmito protokolmi. Štruktúra serveru a formát dát, potrebných pre fungovanie systému, na lokálnom a SFTP repozitári budú zhodné.



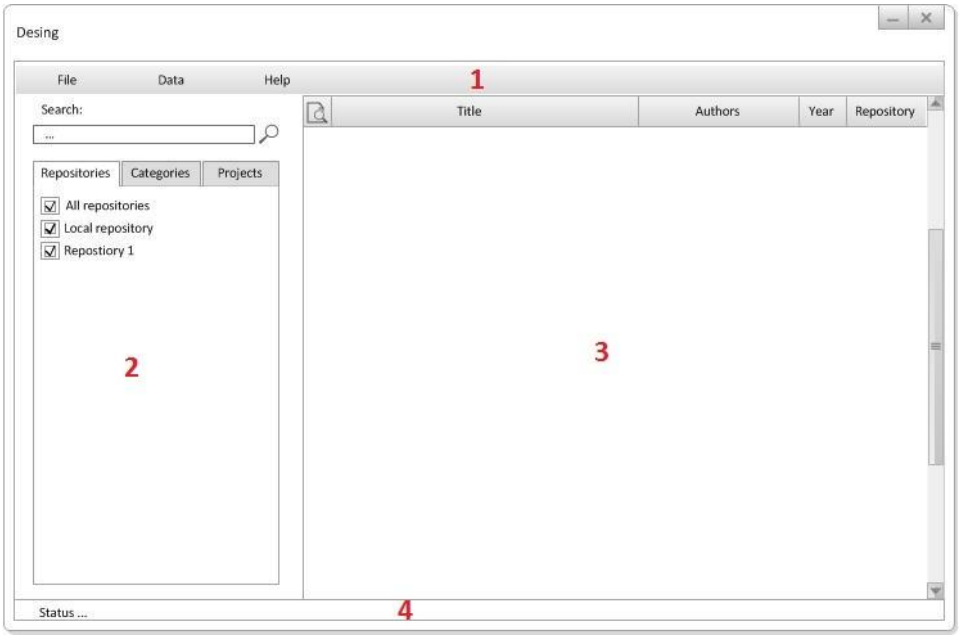
Obrázok 4.1 Znázornenie systému

4.2 Klient – užívateľské rozhranie

Klient bude grafická aplikácia, ktorá ma užívateľovi poskytnúť rýchly, jednoduchý a prehľadný prístup k dokumentom uložených v systéme i mimo neho, v internetových databázach. Pre aplikácie s grafickým užívateľským rozhraním je typická jednoduchosť a prehľadnosť aplikácie a taktiež jednoduchosť a intuitívnosť ovládania. S ohľadom na to a na poznatky získané z analýzy už existujúcich aplikácii som sa snažil navrhnúť aj užívateľské rozhranie mojej aplikácie.

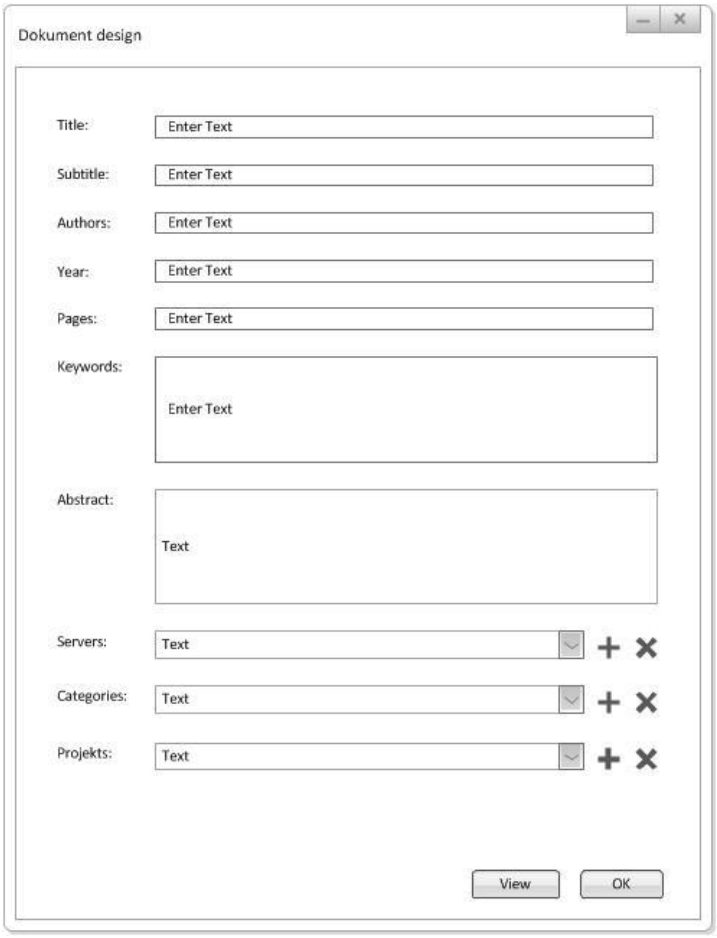
Program bude obsahovať dve základné okná. Hlavné okno (návrh je možné vidieť na obrázku 4.2) a okno na vytváranie a editáciu dokumentov (obrázok 4.3). Hlavné okno môžeme rozdeliť do štyroch základných častí:

1. Ovládací panel s rôznymi položkami, pričom každá táto položka bude obsahovať roletové menu združujúce operácie s logicky súvisiacou funkcionalitou.
2. Panel pre vyhľadávanie, ktorý obsahuje text box pre vyhľadávanie. Ďalej sa tu nachádza panel s možnosťou nastaviť, v ktorom repozitári chceme vyhľadávať, panel zobrazujúci strom kategórií a panel zobrazujúci strom projektov. V posledných dvoch spomenutých paneloch je možné kliknutím na kategóriu/projekt zobraziť všetky dokumenty obsiahnuté v týchto kategóriách/projektoch.
3. Panel slúžiaci na zobrazovanie výsledkov vyhľadávania. Výsledky sa budú zobrazovať v tabuľke do riadkov pod sebou. Každý riadok bude zobrazovať typ dokumentu, názov dokumentu, autora, rok vydania, server, na ktorom sa dokument nachádza. Po kliknutí na typ dokumentu sa daný súbor otvorí v príslušnom programe na prehliadanie. Po dvoj kliknutí na riadok sa zobrazí okno s podrobným výpisom informácií o danom dokumente s možnosťou upravovať tieto informácie (ak je to možné).



Obrázok 4.2 Návrh hlavného okna

Okno na editáciu dokumentov bude jednoduché, informatívneho charakteru. Bude zobrazovať všetky dostupné informácie o dokumente v databáze a umožňovať ich editáciu. V spodnej časti sa budú nachádzať tlačítko View pre zobrazenie dokumentu v odpovedajúcom programe a tlačítko OK, ktorým sa okno uzavrie.



Obrázok 4.3 Návrh okna dokumentu

4.3 Návrh databázy

Dáta na lokálnom úložisku a na SFTP serveroch budú uložené v databáze MongoDB (2.1) databáze. Informácie, ktoré bude databáza o jednotlivých dokumentoch uchovávať sú nasledujúce:

- ID dokumentu (je generované automaticky databázou)
- nadpis dokumentu
- podnadpis dokumentu
- autor dokumentu (v prípade že ich je viac, bude každý zapísaný ako samostatná položka)
- rok vydania
- počet strán dokumentu

- kľúčové slová
- abstrakt
- štítky (budú špecifické pre každého užívateľa)
- server (na ktorom je dokument uložený)
- typ servera
- kategórie (jeden dokument sa môže nachádzať vo viacerých kategóriách)
- projekty (jeden dokument sa môže nachádzať vo viacerých projektoch
- meno užívateľa, ktorý dokument vložil do databázy
- dátum a čas vloženia do databázy
- rok vloženia
- typ dokumentu
- súbory

Podobu záznamu jedného dokumentu môžeme vidieť na obrázku 4.4. Položky „authors“, „categories“ a „projects“ sú polia typu „string“. Položka „doc_date“ je typu „long“. Položka „labels“ je pole, kde jeden záznam je reprezentovaný dvojicou užívateľ : hodnota. Položka „files“ je pole objektov uchovávajúcich si dáta k súborom. Sú to:

- názov súboru (aj s príponou typu súboru)
- názov súboru uloženého v repozitári
- veľkosť súboru
- hodnota typu „boolean“ indikujúca, či je súbor webový odkaz

4.4 Komunikácia s HTTP serverom

Implementácia HTTP protokolu súvisí s možnosťou komunikovať so vzdialenými on-line databázami. To obmedzuje aj množstvo operácií, ktoré môžeme nad týmito repozitármi vykonávať:

- vyhľadávanie
- kopírovať dokument

Keďže formát požiadavku a následnej odpovede na internetové databáze sa líši od servera k serveru, nie je možné vytvoriť jednotné komunikačné rozhranie. Preto samotný program nebude priamo implementovať komunikáciu pomocou tohto protokolu. Namiesto toho bude mať možnosť spúšťať externé skripty napísané v programovacom jazyku Jython (2.4). V tomto skripte bude nutné, či už s využitím modulov jazyka Jython alebo knižníc jazyka Java, doprogramovať komunikáciu s HTTP serverom, vytvoriť požiadavku na vyhľadávanie, ktorá sa na server odošle a následné spracovanie odpovede do preddefinovaných štruktúr. Získané dáta potom spracuje už samotná aplikácia, ktorá ich

vloží do databázy pre ich ďalšie použitie. Čiže pre každý on-line repozitár bude potrebné naprogramovať plugin[11].


```
{
   "_id": {
      "$oid": "4faa40c2585f61aff1311fc8"
   },
   "server": "Merlin",
   "title": "Bakalárska práca",
   "subtitle": "",
   "authors": [
      "Juraj Paška"
   ],
   "pages": "30",
   "year": "2012",
   "keywords": "bakalárska praca, MongoDB, Java",
   "abstract": "Práca o systéme pre správu vedeckých publikácií",
   "categories": [
      "Java/Jython"
   ],
   "projects": [
      "Bakalárka"
   ],
   "user": "Kokso",
   "doc_date": 1336557762438,
   "doc_year": "2012",
   "files": [
      {
         "dbFileName": "Merlin4faa40c2585f61aff1311fc8",
         "fileName": "bakalarka.pdf",
         "isLink": false,
         "filesize": 3458
      }
   ],
   "labels": [
      {
         "user": "Kokso",
         "label": "Read"
      }
   ]
}
```


Obrázok 4.4 Príklad záznamu v databáze

[11] Plugin alebo tiež zásuvný modul je program, ktorý nepracuje samostatne, ale ako doplnkový modul inej aplikácie rozširuje jej funkčnosť.

4.5 Komunikácia s SFTP a lokálnym serverom

Ako som už vyššie spomínal, štruktúra komunikácia s SFTP a lokálnym serverom bude rovnaká. Na každom takomto serveri by mala byť uložená v špecifickom formáte reprezentácia databázy. Zvolil som si súbor typu JSON, keďže databáza, ktorú sme sa rozhodli používať na klientovi (MongoDB) dokáže jednoducho exportovať a importovať svoje dáta práve do tohto formátu. Ďalej sa tam bude nachádzať súbor (ďalej *hash*), ktorého úlohou je informovať, že databáza bola zmenená, a je potrebné ju v klientskej aplikácii aktualizovať. A posledným súborom, dôležitým pre správne fungovanie systému, bude súbor nazvaný *lock*, ktorý má slúžiť pre riadenie prístupu viacerých užívateľov, s cieľom zápisu. Výsledná štruktúra repozitáru bude preto nasledovná:

- súbor/súbory s koncovkou *hash*
- súbor/súbory s koncovkou *json*
- súbor s názvom *lock*
- adresár s názvom *files*, v ktorom sa budú nachádzať samotné súbory jednotlivých dokumentov

Aplikácia bude môcť so súbormi uloženými na týchto serveroch vykonávať nasledujúce operácie:

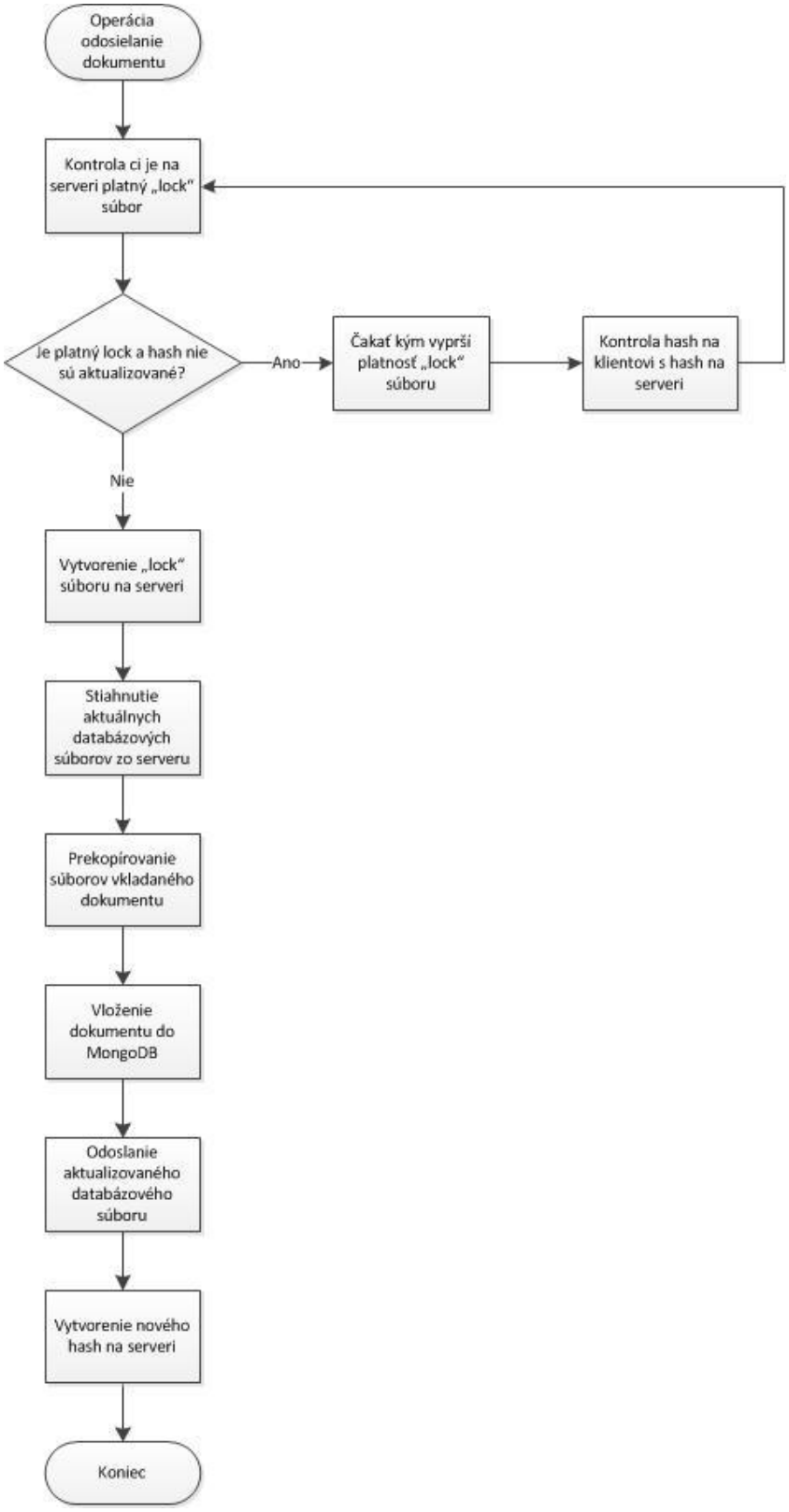
- vyhľadávanie
- vložiť nový dokument
- zmazať dokument
- kopírovať dokument (nebude možné na HTTP server)
- presunúť dokument (nebude možné na HTTP server)
- upraviť dokument

4.5.1 Proces zmeny dát na serveri

Pri úprave dát v databáze (rozumej na serveri) je potrebné dbať na možnosť prístupu viacerých užívateľov k danému repozitáru. Priebeh tejto činnosti je možné vidieť na obrázku 4.5, ktorý znázorňuje proces vloženia dokumentu na server:

1. Po pripojení k serveru sa stiahne súbor s názvom *lock* a skontroluje sa či je platný. To znamená, či niekto iný v danom momente neodosiela dáta na server. V prípade, že tomu tak je, tak aplikácia na základe dát získaných zo súboru pozastaví proces. Takisto prebehne kontrola, či už prebehla kontrola aktuálnosti dát (krok 2). V prípade že áno, pokračuje sa na krok 3.
2. Nasleduje kontrola súborov s *hashom*. Jej úlohou je zistiť, či pracujeme s aktuálnymi dátami. V prípade že sú *hashe* (*hash* na klientovi s *hashom* na serveri) rozdielne, aplikácia to zaznamená a neskôr (krok 4) vykoná aktualizáciu. Nasleduje opätovná kontrola *lock* (krok 1).
3. Vytvorenie súboru *lock*. Tým sa zabezpečí výlučný prístup.

4. Aktualizácia databázy aplikácie, odoslanie súborov, odoslanie upravenej databázy, vytvorenie nového *hash*.



Obrázok 4.5 Vloženie dokumentu na server

5 Implementácia

Program Repo, ktorý je výsledkom tejto bakalárskej práce, je grafická aplikácia napísaná v jazyku Java s využitím grafickej knižnice Swing. Bol vyvíjaný pod operačným systémom Windows za pomoci vývojárskeho nástroja Netbeans 7.1 s verziou JDK 7. Keďže je Java multiplatformný programovací jazyk, nie je problém spustiť aplikáciu tak pod Windows, ako aj pod operačným systémom Linux. Jediný rozdiel je v spúšťaní databázy, ktorá sa v systéme Windows púšťa automaticky (v prípade že už nie je pustená) pri štarte aplikácie. V Linuxe je nutné pustiť databázu manuálne pred spustením samotnej aplikácie.

Program môžeme rozdeliť do troch častí, kde každá časť je reprezentovaná balíkom. A to balíkom `repo` a jeho dvoma podbalíkmi `repo.database` a `repo.communication`. Hlavný balík `repo` zastrešuje celé užívateľské prostredie a jeho ovládanie, nachádza sa tu taktiež hlavná trieda MainWindow, v ktorej prebieha inicializácia všetkých dôležitých objektov programu. Balík `repo.database` sa stará o komunikáciou s databázou MongoDB a `repo.communication` zaobstaráva komunikáciu s lokálnym a SFTP serverom, alebo v prípade HTTP serveru, obstaráva spracovanie skriptov.

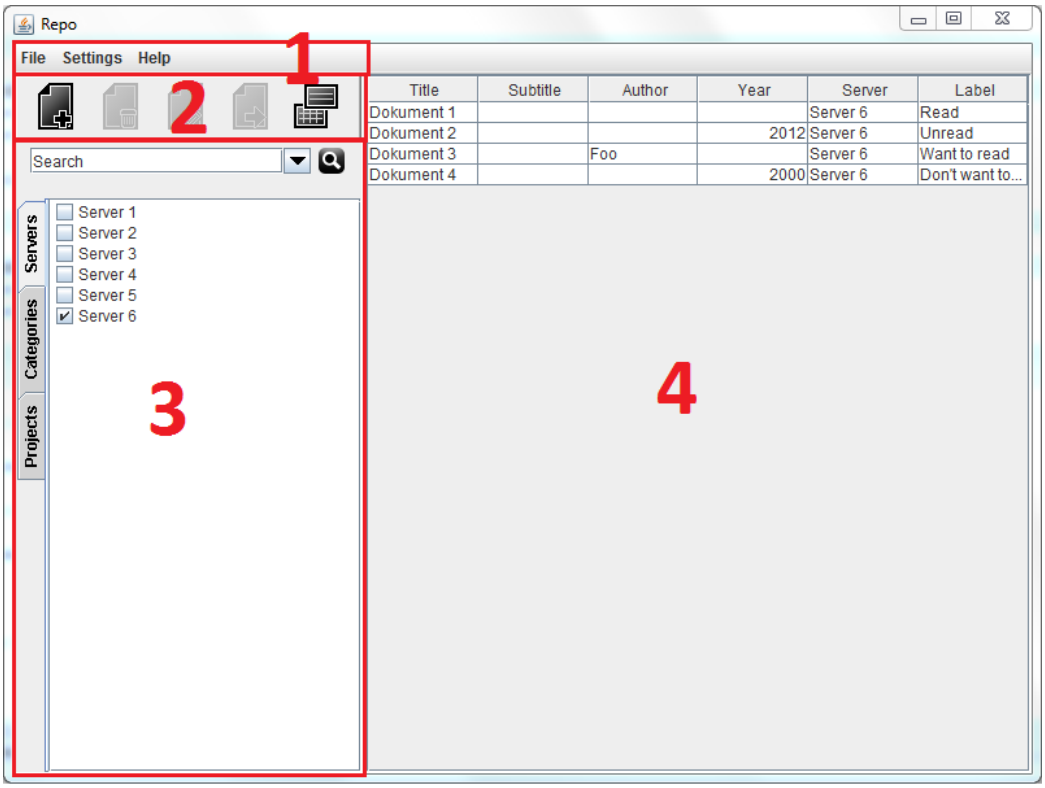
5.1 Grafické rozhranie

Program je implementovaný ako desktopová aplikácia s využitím knižnice Swing (obrázok 5.1). Jeho implementácia sa nachádza v triede `MainWindow`. Rozloženie ovládacích prvkov na ploche sa vo všeobecnosti nelíši od ostatných podobných aplikácií (viď 3.3).

V hornej časti (1) sa nachádza klasická lišta s menu, v ktorých sa nenachádzajú žiadne ovládacie prvky, ale funkčné nastavenia a prvky informačného charakteru. Zvyšok okna je rozdelený pomocou komponenty `JSplitPane` na dve časti tak, aby ich mohol užívateľ podľa potreby rozširovať alebo zužovať.

Vo vrchnej ľavej časti sa nachádza panel (2) so štyrmi tlačítkami pre manipuláciu s dokumentmi a zatiaľ neimplementované tlačítko pre zmenu vzhľadu zobrazovacej časti z tabuľky na citácie (viď I, Librarian). Pod týmto panelom sa nachádza časť pre organizáciu a vyhľadávanie dát (3). V nej je textové pole pre zadávanie textu na vyhľadávanie. A pod ním komponenta `JTabbedPane`, ktorej vzhľad je pomocou triedy `VerticalTextIcon` pozmenený tak, aby sa názvy jednotlivých záložiek panela vypisovali vo zvislom smere. V prvej záložke sa nachádza zoznam `JList` upravený pomocou triedy `RepoCheckList` na zoznam `JCheckBoxov`. V ďalších dvoch záložkách je obsiahnutá komponenta `JTree`, o ktorej správu sa stará trieda `RepoTree`.

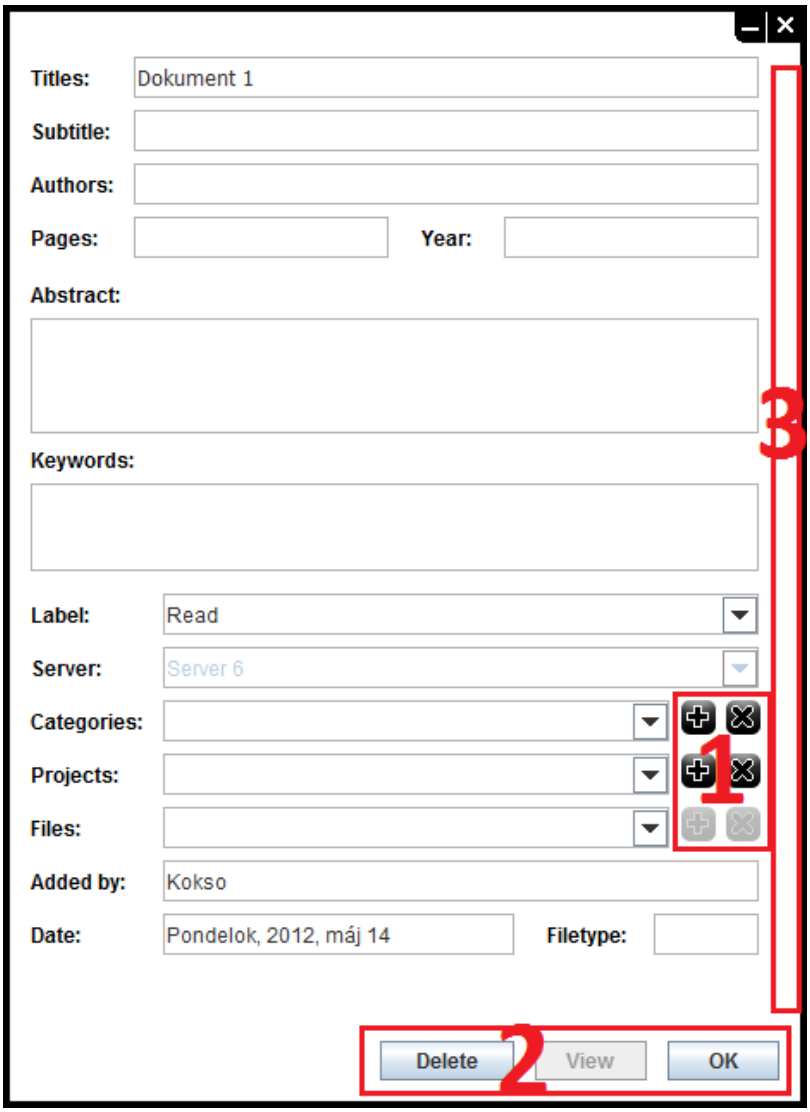
Zvyšok okna, vypĺňa tabuľka (4) pre zobrazovanie nájdených alebo vyfiltrovaných dát. O prístup k nej a jej vyplňovanie sa stará trieda `RepoTree`. Prístup k tabuľke je synchronizovaný z dôvodu súčasného prístupu viacerých vlákien pri zobrazovaní nájdených dát (viď).



Obrázok 5.1 Grafické rozhranie aplikácie

5.1.1 Dokument

Druhou veľmi dôležitou časťou aplikácie z pohľadu grafického rozhrania je okno zobrazujúce detailný náhľad na dokument (obrázok 5.2). Tvorí ho jednoduché okno v podobe bieleho listu. Dáta sú prevažne vypisované/zapisované do klasických textových polí typu `JTextField`. Položky *Abstract* a *Keywords* sú typu `JTextArea` a v prípade, že text zapisovaný do týchto dvoch polí presiahne ich veľkosť, polia sa budú postupne zväčšovať a na pravej strane okna dokumentu (3) sa zobrazí skrolovacia lišta. Ostatné dáta, ktoré sa nevypisujú do textových polí, využívajú editovateľné alebo needitovateľné komponenty typu `JComboBox`. Pri položkách Categories, Projects a Files sa nachádzajú tlačítka (1), ktoré majú schopnosť pridávať alebo odoberať položky z `JComboBoxov`. V spodnej časti okna (2) sa nachádzajú tlačítka *Delete* (zmazanie zobrazeného dokumentu) a *View* (zobrazenie vybraného súboru dokumentu), ktoré sú viditeľné len pri zobrazení existujúceho dokumentu, nie pri tvorbe nového. Tlačítko *OK* je viditeľné v oboch prípadoch.



Obrázok 5.2 Náhľad dokumentu

5.2 Štruktúra servera

Aplikácia dokáže ukladať dokumenty tak ako na miestnom úložisku dokumentov v súborovom systéme, tak aj na vzdialenom prostredníctvom SFTP spojenia. Úložisko (repozitár) obsahuje adresár *files*, v ktorom sa nachádzajú súbory dokumentov. Ďalej sa tam nachádzajú súbory typu JSON s dátami o dokumentoch, ktoré reprezentujú obsah databázy. V jednom repozitári sa môže nachádzať viac takýchto súborov. Názov JSON súboru je zložený z *roku* a prípony *.json*, napr:

```
2012.json
```

Rok v názve súboru sa vzťahuje na rok uloženia dokumentu do databázy. Čiže v súbore z vyššie uvedeného príkladu budú len záznamy dokumentov vložených v roku 2012.

Každému JSON súboru odpovedá jeden súbor s príponou *.hash*. Ten slúži na identifikáciu zmeny v príslušnom JSON súbore. Mennú časť tohto súboru tvorí takisto ako u JSON súboru, ktorému odpovedá, *rok*. „Hash“ súbor obsahuje len jeden riadok, ktorého obsahom je hash. Ten je tvorený z názvu súboru spojeného s dátumom vytvorenia „hash“ súboru vo forme čísla typu long:

```
Názov_serveru1336734266799
```

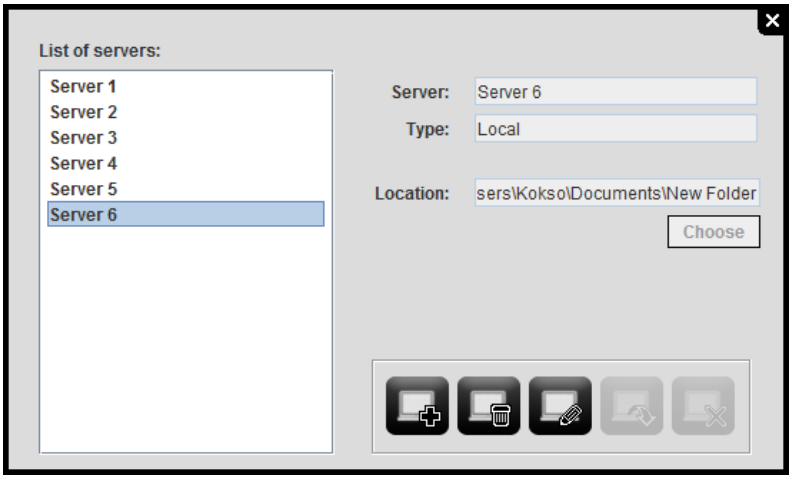
S ním si aplikácia porovnáva hash uložený u seba.

Ďalším súborom nachádzajúcom sa na serveri je súbor s názvom *lock*. Obsahuje dva riadky. Na prvom je čas poslednej modifikácie súboru, opäť vo forme čísla long. Na druhom riadku je v milisekundách uvedený čas, ktorý určuje dobu od poslednej modifikácie, po ktorú nesmie iná aplikácia upravovať súbory na serveri. Obsah súboru vyzerá nasledovne:

```
1336734266784
5000
```

5.3 Server v aplikácii

Server v programe je reprezentovaný abstraktným objektom `Server`. Uchováva si dve informácie spoločné pre všetky tri typy serverov. A to názov serveru, ktorý musí byť jedinečný a typ serveru. Servery je možne pridávať, mazať, upravovať.



Obrázok 5.3 Správa serverov

5.3.1 SFTP a lokálny server

Ďalšie informácie, ktoré si servery ukladajú, sú závislé od typu servera. Lokálny server, reprezentovaný triedou `LocalServer`, si ukladá umiestnenie repozitára na disku. SFTP server, reprezentovaný triedou `SFTPServer`, si ukladá adresu servera a prihlasovacie údaje k nemu. Keďže oba tieto servery sa pripájajú na rovnaké typy repozitára, len odlišným spôsobom, obsahujú aj

rovnakú sadu metód na prácu s dokumentmi (viď operácie). Táto sada metód je definovaná rozhraním `ServerMethods`, ktoré triedy týchto typov serverov implementujú. Všetky funkcie vykonávajú svoju činnosť v samostatných vláknach (platí aj pre HTTP server).

5.3.2 HTTP server

Server HTTP, reprezentovaný triedou `HTTPServer`, je primárne určený pre komunikáciu s internetovými databázami, čo obmedzuje aj množstvo funkcií, ktoré podporuje, na dve: vyhľadávanie a kopírovanie dokumentov zo serveru (nie na HTTP server).

Samotná trieda komunikáciu cez HTTP protokol nevykonáva, ale implementuje používanie skriptov písaných v jazyku Jython vo forme pluginov (vzor pre písanie Jython skriptov sa nachádza v adresári projektu *./scripts/template.py*). Program volá zo skriptu funkciu `search`, do ktorej vo forme parametrov posiela reťazec pre vyhľadávanie a nastavenia vyhľadávania. Funkcia `search` zase vracia naspäť pole objektov typu `Data`, ktoré následne vloží do MongoDB databázy, kde s nimi môže ďalej pracovať.

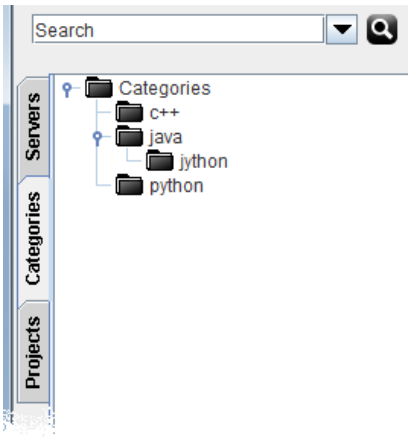
Výhodou tohto prístupu je, že samotné vyhľadávanie nie je odkázané čisto na HTTP protokol, ale je ho možné implementovať akýmkoľvek spôsobom.

5.4 Vyhľadávanie

Vyhľadávanie je jednou z najdôležitejších funkcií aplikácie pre správu vedeckých publikácií. Pre skrátenie doby vyhľadávania prebieha vyhľadávanie paralelne pre každý vybratý server (obrázok 5.4). Nájdené dokumenty sa následne uložia do databázy a postupne sa vykreslia do tabuľky, tak ako servery budú ukončovať vyhľadávanie. Tieto dáta (z posledného vyhľadávania) je možné v tabuľke zoraďovať, alebo ich filtrovať pomocou stromu kategórií a projektov (obrázok 5.5) bez toho, aby sa dáta opätovne museli vyhľadávať na serveroch.

Počas procesu vyhľadávania sú tabuľka, operácie s dokumentmi a samotné vyhľadávanie neprístupné, až kým všetky servery neukončia svoju činnosť. Po ukončení vyhľadávania alebo aj akejkoľvek inej činnosti bežiacej v samostatnom vlákne, sa pošle notifikácia objektu typu `Monitor`, ktorý implementuje rozhranie `Observer`. Ten počká, kým ukončia činnosť všetky vlákna a sprístupní všetky blokované funkcie.

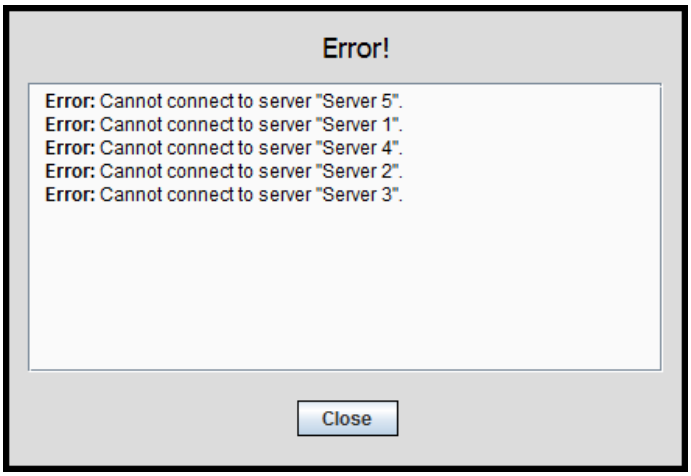
Obrázok 5.4 Servery



Obrázok 5.5 Kategórie a projekty

5.5 Chybové pripojenia

V prípade, že sa nepodarí aplikácii pripojiť na server, oznámi to daný server, ktorý toto pripojenie vykonával, objektu typu `Monitor`. Ten po ukončení činnosti všetkých vlákien určených na komunikáciu so servermi vypíše zoznam neúspešných pripojení (obrázok).



Obrázok 5.6 Chybové hlásenie

6 Testovanie

Aplikácia bola testovaná na počítači s operačným systémom Windows s Javou vo verzii 7. Vzdialené úložisko, pre testovanie SFTP komunikácie, bolo umiestnené na zariadení s operačným systémom Linux.

Nasledujúce príklady testov boli vykonávane ako na lokálnom úložisku (rozumej test triedy `LocalServer`), tak aj na vzdialenom úložisku (rozumej test triedy `SFTPServer`):

Testovanie aktualizácie databázy klienta – na strane servera (viď 5.2) som manuálne upravil súbor s hashom. Takisto som upravil záznam v súbore JSON, aby som mohol skontrolovať, či nastala očakávaná zmena dát pri výpise na strane klienta.

Testovanie odosielania dokumentu na server – zaznamenal som si stav súborov JSON, hash, lock a obsah adresára files na strane servera. Po prevedení operácie odosielania som skontroloval, či boli všetky 3 zmenené, v súbore JSON som skontroloval, či tam pribudol záznam o vkladanom dokumente a v adresári files som vyhľadal súbor, ktorý patril k odoslanému dokumentu.

Testovanie zmazania dokumentu – podobne ako v predchádzajúcom prípade som si zaznamenal stav súborov a adresára files na serveri. Po prevedení operácie som opäť skontroloval, či boli všetky 3 súbory zmenené, či bol zo súboru JSON odstránený korektný záznam a či bol z adresára files zmazaný správny súbor.

Testovanie súbežného prístupu 2 klientov s cieľom vložiť dokument – zaznamenal som si stav súborov na strane servera. Z klienta 1 som spustil operáciu odosielania dokumentu. Hneď na to som pustil rovnakú operáciu z klienta 2. Prvá vec, ktorú som kontroloval, bolo, či klient 2 pozastavil odosielanie dokumentu do doby stanovenej klientom 1. Po splnení tohto stavu, som skontroloval či boli súbory z oboch klientov správne uložené na server.

Druhá časť testovania bola zameraná na funkčnosť grafického rozhrania aplikácie. Aplikácia bola poskytnutá ďalším osobám. V tejto časti mi pomohli môj vedúci práce, Ing. Jan Kněžík a môj známy, Bc. Peter Zubčák. Otestovali a poskytli mi informácie ohľadom chýb týkajúcich sa funkčnosti grafických komponent v špecifických situáciách, plus názor na komfortnosť ovládania a vizuálny dojem aplikácie.

7 Záver

Pri vypracovávaní tejto bakalárskej práce som mal možnosť získať nové vedomosti v oblasti dokumentovo orientovaných databáz, naučil som sa pracovať s databázovým systémom MongoDB a vyskúšal som si integráciu jedného programovacieho jazyka do druhého.

Aplikácia bola navrhnutá a implementovaná ako viac užívateľský systém umožňujúci správu vedeckých publikácií s možnosťou pracovať so vzdialenými úložiskami dokumentov prostredníctvom protokolov SFTP a HTTP. Implementáciou rozšíriteľnosti aplikácie prostredníctvom externých skriptov som nechal otvorenú možnosť pre prácu s úložiskami dát komunikujúcich s využitím aj iných protokolov.

Za čiastočnú nevýhodu systému by som označil neexistenciu serverovej aplikácie, ktorá by riadila prístup užívateľov k dokumentom uložených na serveri. Na druhej strane treba ale oceniť schopnosť jednoducho zriadiť repozitár, bez nutnosti inštalácie ďalšieho softwaru.

Aplikácia má pravdaže priestor na rozširovanie svojej funkčnosti. Ako prvé možné rozšírenie a pravdepodobne aj najdôležitejšie z pohľadu ďalšieho rozvoja a šírenia aplikácie sa javí možnosť vyhľadávať v čo najväčšom množstve internetových databáz. Výhodou je, že aplikácia je na takúto formu rozšírenia pripravená a nie je nutné meniť jej kód.

Ďalším z rozšírení by mohla byť možnosť zobrazovať dáta, okrem tabuľky, aj vo forme citácií, možnosť exportovania a importovania dokumentov do súborov typu CSV, RIS , EndNote, BibTex a iných. Vylepšením by určite bola možnosť extrakcie metadát zo súborov vo formáte PDF a generovanie citácií z informácií uložených o dokumente.

8 Literatúra

[1] Affero General Public License. In: *Wikipedia: the free encyclopedia* [online]. 2001- [cit. 2012-05-06]. Dostupné z: http://en.wikipedia.org/wiki/Affero_General_Public_License

[2] DIROLF, Kristina Chodorow and Michael a [foreword by Jeremy ZAWODNY]. *MongoDB:the definitive guide*. 1st ed. Beijing: O'Reilly, 2010. ISBN 978-1-449-38156-1.

[3] *BSON* [online]. [cit. 2012-05-06]. Dostupné z: http://bsonspec.org/

[4] *Memcached* [online]. 2001- [cit. 2012-05-06]. Dostupné z: http://en.wikipedia.org/wiki/Memcached

[5] *MongoDB* [online]. [cit. 2012-05-06]. Dostupné z: http://www.mongodb.org

[6] MongoDb vs SQL Server Basic Speed Tests. *EggHeadCafe* [online]. 2010-08-29 [cit. 2012-05-06]. Dostupné z: http://www.eggheadcafe.com/tutorials/database/6f573869-c8eb-40c3-9946-2f61e0163966/mongodb-vs-sql-server-basic-speed-tests.aspx

[7] Insert performance comparison of NoSQL vs SQL servers. *Artur Ejsmont* [online]. 2011 [cit. 2012-05-06]. Dostupné z: http://artur.ejsmont.org/blog/content/insert-performance-comparison-of-nosql-vs-sql-servers

[8] *Introducing JSON* [online]. [cit. 2012-05-06]. Dostupné z: http://www.json.org/

[9] HEROUT, Pavel. *Učebnice jazyka Java*. 1. vyd. České Budějovice: Kopp, 2001, 349 s. ISBN 80-723-2115-3.

[10] Java (programming language). In: *Wikipedia*: the free encyclopedia [online]. 2001- [cit. 2012-05-06]. Dostupné z: http://en.wikipedia.org/wiki/Java_(programming_language)

[11] BILL, Robert W. *Jython for Java programmers*. 1st ed. Indianapolis, Ind.: New Riders, 2002, 466 s. ISBN 07-357-1111-9.

[12] JUNEAU, Josh, Jim BAKER, Victor NG, Leo SOTO a Frank WIERZBICKI. *The definitive guide to Jython: Python for the Java platform*. New York: distributed by Springer-Verlag, 2009, 511 s. ISBN 14-302-2527-0.

[13] Jython User Guide. *The Python Wiki* [online]. 2011-04-25 [cit. 2012-05-07]. Dostupné z: http://wiki.python.org/jython/UserGuide

[14] Scripting Jython with JDK 6. *The Python Wiki* [online]. 2008-11-15 [cit. 2012-05-07]. Dostupné z: http://wiki.python.org/jython/JythonMonthly/Articles/October2006/1

[15] SSH File Transfer Protocol. IETF Tools [online]. 2007-01-11 [cit. 2012-05-07]. Dostupné z: http://tools.ietf.org/html/draft-ietf-secsh-filexfer-13

[16] The Secure Shell (SSH) Protocol Architecture. IETF Tools [online]. 2006-01 [cit. 2012-05-07]. Dostupné z: http://tools.ietf.org/html/rfc4251

A Obsah priloženého CD média

Obsahom sú nasledujúce súbory a adresáre:

`./Bakalárska_práca.pdf`
- Elektronická podoba bakalárskej práce

`./README`
- Návod na inštaláciu

`./Java/`
- Inštalácia Javy

`./Jython/`
- Inštalácia Jython

`./Repo/`
- Samotná aplikácia

`./Repo/Repo.jar`
- Spúšťací súbor aplikácie

`./Repo/build.xml`
- Skript na kompiláciu zdrojových súborov pomocou Ant

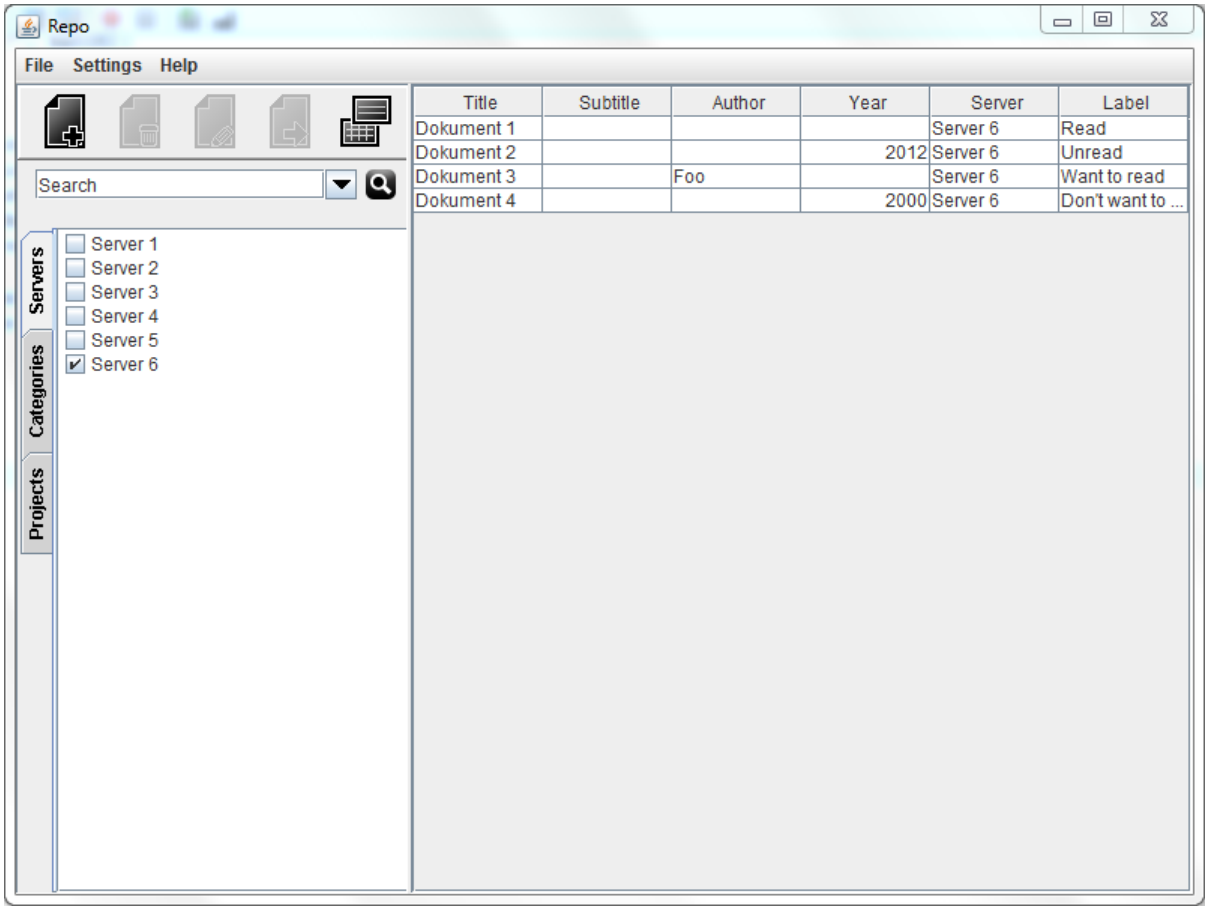
`./Repo/...`
- Zvyšok adresára tvoria súbory potrebné pre chod aplikácie, nápoveda, javadoc, MongoDB databáza

B Manuál

B.1 Užívateľské prostredie

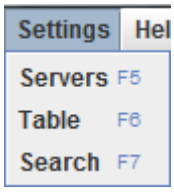
Užívateľské prostredie hlavného okna je možné rozdeliť do 3 základných častí:

- Horné menu s nastaveniami a nápovedou
- Ovládací panel, kde sa nachádzajú navrchu tlačítka pre prácu s dokumentami, pod nimi je vyhľadávacie okno a pod nimi sa nachádza navigačný panel so zoznamom projektov, kategórií a serverov
- Zobrazovacia plocha, kde sa zobrazujú výsledky vyhľadávania



B.2 Nastavenia

Nastavenia sa nachádzajú v menu "Settings" obsahuje 3 položky:



- Servers - otvorí okno na pridávanie, odstraňovanie a editáciu serverov.
- Table - otvorí okno pre nastavenie tabuľky, ktoré dáta sa majú zobrazovať.
- Search - zobrazí okno s možnosťou nastavenia, v ktorých dátach sa ma vyhľadávať. V základe sú nastavené keywords (kľúčové slová).

B.3 Práca s dokumentmi

S dokumentmi sa dajú robiť 4 základné operácie:

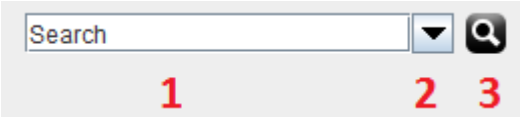
Vytvoriť nový dokument

Zmazať existujúci dokument

Editovať/Zobraziť detaily o dokumente (taktiež je možné túto akciu vyvolať dvojklikom na položku v tabuľke)

Kopírovať/Presunúť existujúci dokument na druhý server

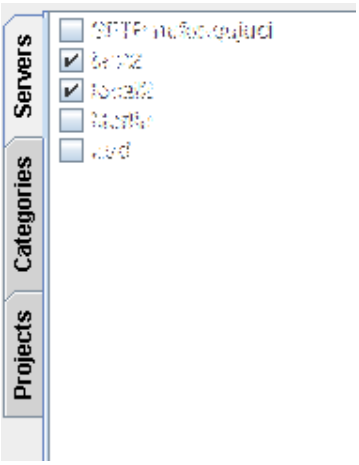
B.4 Vyhľadávanie



Vyhľadávacia časť obsahuje 3 prvky:

- Textové pole, do ktorého sa zadáva výraz pre výhľadávanie
- Tlačítko pre zobrazenie nastavení výhľadávania
- Tlačítko vyhľadávať

B.5 Navigačný panel



Navigačný panel pozostáva z 3 častí:

- Panel Servers, ktorý obsahuje zoznam serveroch a v ktorom je možné označit, na ktorých serveroch sa má vyhľadávať
- Panel Categories, ktorý v podobe stromu zobrazuje zoznam kategorií pre označené servery
- Panel Projects, ktorý v podobe stromu zobrazuje zoznam projektov pre označené servery

Po dvojkliknutí na konkrétnu kategóriu (alebo projekt), sa v tabuľke zobrazia len tie záznamy, ktoré patria do danej kategórie (projektu). Pre opätovné zobrazenie všetkých pôvodne vyhľadaných záznamov stači buď znovu spustiť vyhľadávanie alebo spraviť dvojklik na koreňový adresár kategorie (Categories) alebo projektu (Projects)